BusinessObjects XI

How to use the RAS SDK .NET with In-process RAS Server

Overview

This technical document discusses how to use the Report Application Server (RAS) .NET SDK without the use of a RAS server. Note that this server-less version of the .NET RAS SDK is available only through the Business Objects OEM Partner program.

Contents

Introduction

The Report Application Server (RAS) SDK is a powerful reporting development tool that allows you to control nearly all aspects of the look and behavior of a Crystal report. This SDK is primarily used for runtime report modification of Crystal reports that are integrated into a multi-tiered BusinessObjects™ Enterprise environment. However, with the release of BusinessObjects Enterprise XI, Business Objects OEM partners have the option of using this SDK in a single-tiered environment.

This technical document discusses how you can use the RAS .NET SDK in a single-tiered BusinessObjects XI environment without the use of a RAS server.

Background

Crystal Reports for .NET SDK

Crystal Reports® for .NET (CR.NET) is the reporting solution of choice for .NET developers. Integrated into Visual Studio .NET 2002, 2003, and 2005, CR.NET provides you with the ability to easily integrate Crystal reports into your .NET applications.

A special edition of Crystal Reports Developer Edition is included with Visual Studio 2003. The Crystal Reports Designer is embedded in the IDE, giving you the ability to create dynamic, graphical reports directly from your development environment. After the report is created, you can then use the Crystal Reports .NET Server Controls to deliver this rich content over the web or through desktop applications to end users. CR.NET is the ideal embedded reporting tool for simple web and desktop reporting solutions.

NOTE	If you are using BusinessObjects Enterprise or Crystal Reports® Server, you are also able to connect your CR.NET reporting applications to the BusinessObjects Enterprise framework. For more information, refer to the technical document Upgrading from Crystal Reports .NET to Enterprise.

Report Application Server .NET SDK

CR.NET allows you to display graphical reports over the web, but what about other functionality such as changing the report's formatting and data source? The ability to programmatically modify or create new Crystal reports is not available in CR.NET, but it is available in the RAS.NET SDK.

RAS.NET includes a report creation API (RCAPI) that allows you to perform actions such as modifying the report's parameters and data source structure, adding new fields and groups, and creating reports.

For example, with CR.NET you are able to add a filter to a report but the filter is temporary and is applied only during the user's viewing session. With RAS.NET, you are able to add a filter to a report and then save that filter permanently to the report.

The RAS SDK offers the following benefits:

- Full access to the database structures in the report using the **DatabaseController**, including detailed connection information.
- Batched XML data exporting and access to the processed data in the report using the **RowsetController** access.
- Full control and access to nearly all objects in a report's subreports using the **SubreportController**.
- Detailed control over printing and export options, including saved printer settings in the report, using the **PrintOutputController**.
- Access to all methods and properties from the **ReportDocument** and **ReportClientDocument** object models if the application creates a **ReportClientDocument** object from the **ReportDocument** object model. Since the **ReportDocument** methods internally call **ReportClientDocument** methods to complete the actions, the two object models remain synchronized.
- Use of the same RAS SDK for stand-alone applications that open reports from a local disk or a managed enterprise environment.
- Ability to programmatically remove and modify nearly any report object. Please consult the RAS SDK documentation for full details on supported objects.
- Detailed support for many objects and properties not accessible in the Crystal Reports for .NET object model, including conditional formulas, the SQL Statement used in the report, and report version information.

NOTE	By using In-process RAS instead of client-server RAS, you forfeit the performance and scalability benefits of a server-oriented out-process system.

Report Application Server object model

The **ReportClientDocument** is the main class in the RAS object model and is found in the **CrystalDecisions.ReportAppServer.ClientDoc** namespace. The ReportClientDocument class represents a Crystal report and is used to access report methods and properties.

When integrated into an enterprise environment, the **ReportClientDocument** object model can be accessed by retrieving the **RASReportFactory** service from the **EnterpriseSession** object. For more information, refer to the section "Binding to the ReportClientDocument Class Instance" in the Business Objects Enterprise XI .NET SDK guide.

For single-tiered RAS applications, you can open a report and access the ReportClientDocument object model by using the **ReportDocument** class's **ReportDocument.ReportClientDocument** property.

For detailed examples on how to use CR.NET, refer to the Crystal Reports XI .NET SDK Guide. RAS SDK documentation is installed as part of the Crystal Reports Server and BusinessObjects Enterprise installations. When installing either of these products, you may choose the SDK-only install. You also have the option of downloading the RAS.NET SDK documentation from our support site.

In-Process Report Application Server

RAS is a client-server technology composed of the RAS SDK (.NET and Java) and the RAS server. Since RAS is an embeddable web-based reporting solution that is server-oriented instead of component-based, the RAS SDK works in conjunction with the RAS server to process and deliver reports.

New in version XI, however, is a server-less version of the RAS technology referred to as In-process RAS. In-process RAS allows the RAS .NET SDK to process reports without a RAS server. The deployment package for an in-process RAS.NET application need only include the .NET assemblies, which are available for download as merge modules from our support site.

Licensing

In-process RAS is available only for the .NET platform and only through the Business Objects OEM Partner Program. It is licensed as a component technology and comes with a default of 3 CPLs (Concurrent Processing License). Contact a Business Objects OEM sales representative for more information on how this component is licensed.

Obtaining the RAS in-process components

The .NET assemblies are included out of the box with Crystal Reports XI but are not installed by default. To install and gain access to the In-process RAS development components, a special OEM Crystal Reports key code is required during installation. This key code may be obtained only through the Business Objects OEM Partner Program.

Sample code

This C# code sample demonstrates how to load a report into the **ReportClientDocument** object and pass parameters to the report using In-process RAS:

```
/* Business Objects C# project references:
CrystalDecisions.Windows.Forms
CrystalDecisions.Shared
CrystalDecisions.ReportSource
CrystalDecisions.ReportAppServer.ClientDoc
CrystalDecisions.ReportAppServer.DataDefModel
CrystalDecisions.ReportAppServer.Controllers
CrystalDecisions.CrystalReports.Engine */

using CrystalDecisions.ReportAppServer.ClientDoc;
using CrystalDecisions.ReportAppServer.Controllers;
using CrystalDecisions.ReportAppServer.DataDefModel;
using CrystalDecisions.CrystalReports.Engine;

public class Form1 : System.Windows.Forms.Form
{
    private CrystalReportViewer crystalReportViewer1;

    // CR.NET document object type
    ReportDocument m_crNetDoc;
    // RAS document object type
    ISCDReportClientDocument m_RasDoc;

    /**************************************************************
    *
    * Before loading the report into the RAS ReportClientDocument
    * type, you have to load the report into a CR.NET
    * ReportDocument type first.
    *
    * Once the ReportDocument object is initialized with the
    * report, you can then retrieve the ISCDReportClientDocument
    * from the ReportDocument.
    *
    * This method loads a Crystal Report into a
    * ReportClientDocument object.
    *
    **************************************************************/
```

```
void CreateReport(string path)
{
    m_crNetDoc = new ReportDocument();

    // Load report into ReportDocument type
    m_crNetDoc.Load(path);

    try
    {
        // Load CR.NET document into ReportClientDocument
        m_RasDoc = m_crNetDoc.ReportClientDocument;
    }
    catch(Exception e)
    {
        System.Console.WriteLine("Error: " + e.Message);
    }

    // In the report, odbc_customer.rpt, there are two
    parameters: Country and Sales.
    // We need to pass 'USA' and 5000 to the parameters
    respectively so that the user does not get prompted
    PassParameter(m_RasDoc, "", 0, "USA");
    PassParameter(m_RasDoc, "", 1, 5000);

    // View the report
    crystalReportViewer1.ReportSource = m_RasDoc.ReportSource;
}

/**************************************************************
* Passes a parameter value to report
*
* PARAMETERS  [in] doc            RAS Report Client Document
*             [in] report_name    subreport name (empty for main)
*             [in] param_name     report parameter name
*             [in] param_value    report parameter value
**************************************************************/

public void PassParameter(
    ReportClientDocument doc,
    string report_name,
    int param_index,
    object param_value)
{
```

```
            string param_name =
            doc.DataDefinition.ParameterFields[param_index].Name;


            // Create parameter discrete value

            ParameterFieldDiscreteValue param_val = new
            ParameterFieldDiscreteValue();


            // Set parameter value

            param_val.Value = param_value;


            // Create parameter value collection

            Values vals = new ValuesClass();


            // Add parameter value to this collection

            vals.Add(param_val);


            // Get DataDefController object from client doc

            DataDefController ddc = doc.DataDefController;


            // Get ParameterFieldController

            ParameterFieldController pc = ddc.ParameterFieldController;

            // Set current value

            pc.SetCurrentValues(report_name, param_name, vals);
        }
    }
```

Additional Resources

Technical Support Site

http://support.businessobjects.com/search/

.NET Merge Modules

http://support.businessobjects.com/mergemodules

Business Objects Enterprise .NET Samples

http://support.businessobjects.com/fix/samplescr.asp?ref=default.asp_selectlist#02

Crystal Reports for .NET SDK Guide

http://support.businessobjects.com/library/docfiles/cps10/downloads/en/crXI_NETSDKGuide_en.zip

BusinessObjects Enterprise XI .NET SDK Guides

http://support.businessobjects.com/library/docfiles/cps10/downloads/en/boeXI_net_docs_en.zip

http://support.businessobjects.com/library/docfiles/cps10/downloads/en/boeXI_net_ServerControls_docs_en.zip

Report Application Server XI .NET SDK Guide

http://support.businessobjects.com/library/docfiles/cps10/downloads/en/rasXI_net_docs_en.zip

Developer Zone for .NET developers
http://www.businessobjects.com/products/dev_zone/net/default.asp